EPH-International Journal of Science And Engineering.
ISSN (Online): 2208-2190
Volume 11 Issue 02 June 2025

DOI: https://doi.org/10.53555/1vhsgg43

HALLUCINATIONS OF THE GENERALIST, BRITTLENESS OF THE EXPERT: A STUDY ON FINE-TUNED OPEN SOURCE LLM FOR SOC ANALYSIS

Abhinav Sharma[1*], Jawahar Thakur[2], T.P. Sharma[3]

[1]Department of Computer Science, Himachal Pradesh University, India*
[2] Department of Computer Science and Engineering, National Institute of Technology, Hamirpur, India
{aasvi2006@gmail.com, jawahar.thakur@hpuniv.ac.in, teek@nith.ac.in}

****Corresponding Author:** Abhinav Sharma*
**Department of Computer Science Himachal Pradesh University aasvi2006@gmail.com*

Abstract– *In the evolving landscape of cyber defense, where knowledge is the decisive edge, the practical deployment of Large Language Models (LLMs) is fraught with challenges. General-purpose models from large providers present significant cost and data privacy barriers for most Security Operations Centers (SOCs), while open-source models often hallucinate in structured domains like MITRE ATT&CK. This study investigates these trade-offs by fine-tuning the DeepSeek R1 Distilled LLaMA 8B model on a curated corpus of MITRE ATT&CK semantics, leveraging the Unsloth framework to demonstrate a reproducible pipeline on consumer-grade GPUs. Notably, the entire fine-tuning process was conducted using commercial Kaggle infrastructure, showing that effective LLM specialization can be achieved without access to high-end hardware. Our evaluation contrasts this fine-tuned "Focused Specialist" with the untuned "Creative Generalist" base model. The results reveal a critical trade-off: the base model, while a superior contextual reasoner, suffers from catastrophic factual hallucinations, making it dangerously unreliable. In contrast, our fine-tuned model achieves 84% exact-match accuracy and proves its trustworthiness, but this specialization introduces a "brittleness" where its knowledge is static and less creatively applied. Our LLM-as-a-Judge evaluation confirmed the fine-tuned model was the superior co-pilot in 73% of test cases, achieving a 25% higher average quality score.*

Keywords: Large Language Models, Cyber Threat Intelligence, MITRE ATT&CK, Fine-Tuning, DeepSeek, Security Operations Center

I. INTRODUCTION

A. Background

The emergence of Large Language Models (LLMs) such as OpenAI's GPT series, Meta's LLaMA, and Google's Gemini has transformed natural language processing (NLP) by enabling advanced capabilities in contextual understanding and structured reasoning [1,2,3]. In cybersecurity, LLMs are increasingly being explored for enhancing workflows within Security Operations Centers (SOCs), where they can support tasks such as automated log analysis, alert enrichment, and incident response. By grounding model outputs in structured ontologies like the MITRE ATT&CK framework, LLMs can help automate critical tasks such as technique classification and analyst action recommendation [4,5,6]. Despite their potential, LLMs face practical obstacles: proprietary models like GPT-4o pose high costs and privacy risks, while smaller open-source models, though deployable locally, often lack cybersecurity alignment—resulting in hallucinations. To address these issues, our work focuses on fine-tuning DeepSeek R1 Distilled LLaMA 8B—an open-source and computationally efficient LLM—on a curated instruction-style dataset derived from the MITRE ATT&CK framework. By employing parameter-efficient tuning (QLoRA) and leveraging the Unsloth framework, we demonstrate a low-cost, privacy-preserving pipeline for adapting LLMs on consumer-grade hardware, making advanced AI accessible for a wider range of security teams.

A.1 Motivation and Problem Statement

Modern Security Information and Event Management (SIEM) systems are foundational to enterprise threat detection, yet they suffer from key limitations. Analysts are frequently overwhelmed by high volumes of low-context alerts, leading to "alert fatigue" where genuine threats are missed [7–11]. Traditional SIEMs also rely heavily on static, rule-based correlation, which is insufficient for detecting nuanced or evolving attack patterns [12–14]. LLMs offer a path to automate alert triage and reasoning, but a critical dilemma emerges. As our research demonstrates, a general-purpose LLM, while a creative reasoner, is prone to dangerous factual hallucinations. This research directly confronts this gap by asking a crucial question: ***"How to build an AI co-pilot that is factually reliable, contextually adaptive, resource-efficient, and privacy-preserving via local deployment?"*** To investigate, we fine-tune DeepSeek R1 Distilled LLaMA 8B using QLoRA and Unsloth, achieving resource-efficient fine-tuning on locally deployable hardware. Our work evaluates the performance trade-offs between the untuned base model and our fine-tuned specialist, ultimately demonstrating that neither model is sufficient on its own. We conclude that a successful deployment requires not only domain-specific fine-tuning but also a sophisticated inference-time architecture to achieve real operational value.

A.2 Objective

The objective of this study is two-fold:

1. To fine-tune an open-source LLM (DeepSeek R1 Distilled LLaMA 8B) on a structured, MITRE ATT&CK-aligned dataset using resource-efficient methods.To critically evaluate the performance trade-offs between the untuned "generalist" model and the fine-tuned "specialist" model on a realistic analyst co-pilot task.

A.3 Contributions

1. **Domain-Specific Fine-Tuning:** We provide a methodology for fine-tuning DeepSeek R1 Distilled LLaMA 8B using 4,096 instruction-style examples derived from the MITRE ATT&CK framework, executed efficiently on consumer-grade GPUs via the Unsloth framework.
2. **Identification of the "Brittle Specialist" Problem:** We demonstrate that while fine-tuning achieves high factual accuracy (e.g., 84% on technique classification vs. 0% for the base model) and eliminates hallucinations, it can produce a model that is "brittle" and fails to apply its knowledge to specific alert contexts.
3. **Comparative Evaluation Framework:** We design and apply a head-to-head, LLM-as-a-Judge (LLMJ) evaluation that reveals the nuanced trade-offs between the base model's superior contextual reasoning and the fine-tuned model's superior factual accuracy.

II. RELATED WORK

A. Use of Large Language Models in Cybersecurity

Large Language Models (LLMs), such as OpenAI's ChatGPT, Meta's LLaMA, and Google's Gemini, have revolutionized natural language processing through advanced language comprehension, contextual reasoning, and structured text generation. Their application in cybersecurity has catalyzed a new wave of innovation in Security Operations Centres (SOCs), where they are now central to tasks such as threat detection, cyber threat intelligence (CTI) analysis, forensic interpretation, and incident response. LLMs are particularly adept at extracting actionable intelligence from unstructured threat reports e.g. CTINexus [17] demonstrates how in-context learning (ICL) enables accurate extraction of malware names, vulnerabilities, and inter-entity relationships—surpassing rule-based and supervised methods even with limited annotation. In code security, models such as GPT-4 have shown superior performance in identifying vulnerabilities compared to traditional static analyzers[5,21]. In SOC environments, LLMs enhance anomaly detection, log analysis, and contextual alert triage. Frameworks like RAGLog and SHIELD integrate retrieval-augmented generation, statistical analysis, and provenance data interpretation to detect APTs and zero-day threats with high precision and low false-positive rates[6,24,27]. LLMs also assist in forensic workflows by summarizing system artifacts, interpreting shell commands, and mapping event sequences to the ATT&CK kill chain[5,20,22,23]. Moreover, LLMs support red and blue team preparedness by simulating realistic attack scenarios for tabletop exercises, reducing planning time, and enabling cost-effective micro-training[4,26].

Despite their potential, LLMs face challenges in cybersecurity, including hallucinations, domain-specific limitations (e.g., NER), resource constraints, prompt sensitivity, and security vulnerabilities [4,5,17,21]. These issues hinder reliable, real-time deployment in SOC environments. To unlock the full potential of LLMs in SOC operations requires domain-aligned fine-tuning, lightweight and scalable architectures, secure deployment practices, and robust evaluation for factual and contextual reliability.

B. Model Overview: Deep Seek -R1 Series

The DeepSeek-R1 series represents a new generation of open-source language models designed with advanced reasoning capabilities. Notably, DeepSeek-R1's performance is comparable to proprietary models such as OpenAI's o1 across a variety of reasoning benchmarks[28,29]. The model used in this study—DeepSeek-R1-Distill-Llama-8B—is a distilled variant of DeepSeek-R1, optimized for performance and efficiency. Distilled using reasoning traces from the larger DeepSeek-R1 model, this 8B parameter model balances cost, accessibility, and reasoning performance, making it suitable for fine-tuning on consumer-grade hardware.

C. Parameter-Efficient Fine-Tuning (PEFT) and Quantization

Training and deploying large-scale language models traditionally imposes significant computational and memory demands, often requiring access to high-end data center infrastructure. To make our research feasible on consumer-grade hardware and align with the practical constraints of many SOCs, we adopted two key optimization strategies: Parameter-Efficient Fine-Tuning (PEFT) and 4-bit Quantization.

C.1 Parameter-Efficient Fine-Tuning (PEFT) with LoRA

Fully fine-tuning an LLM requires updating all of its billions of parameters, a process that consumes enormous memory and computational resources. PEFT methods overcome this challenge by freezing the vast majority of the pre-trained model's weights and only training a small number of additional parameters [30, 31].

For our implementation, we selected LoRA (Low-Rank Adaptation), one of the most effective and widely adopted PEFT techniques (Fig. I.). As a visual metaphor, if the pre-trained model is a complex, pre-calibrated lens system, LoRA adds small, lightweight "filters" at strategic points. These filters subtly redirect the model's focus toward our specific task—in this case, understanding MITRE ATT&CK semantics—without altering the core optical system. In our work, LoRA adapters constituted approximately 0.52% of the total model parameters (~41.9 million), enabling effective adaptation with minimal resource overhead.

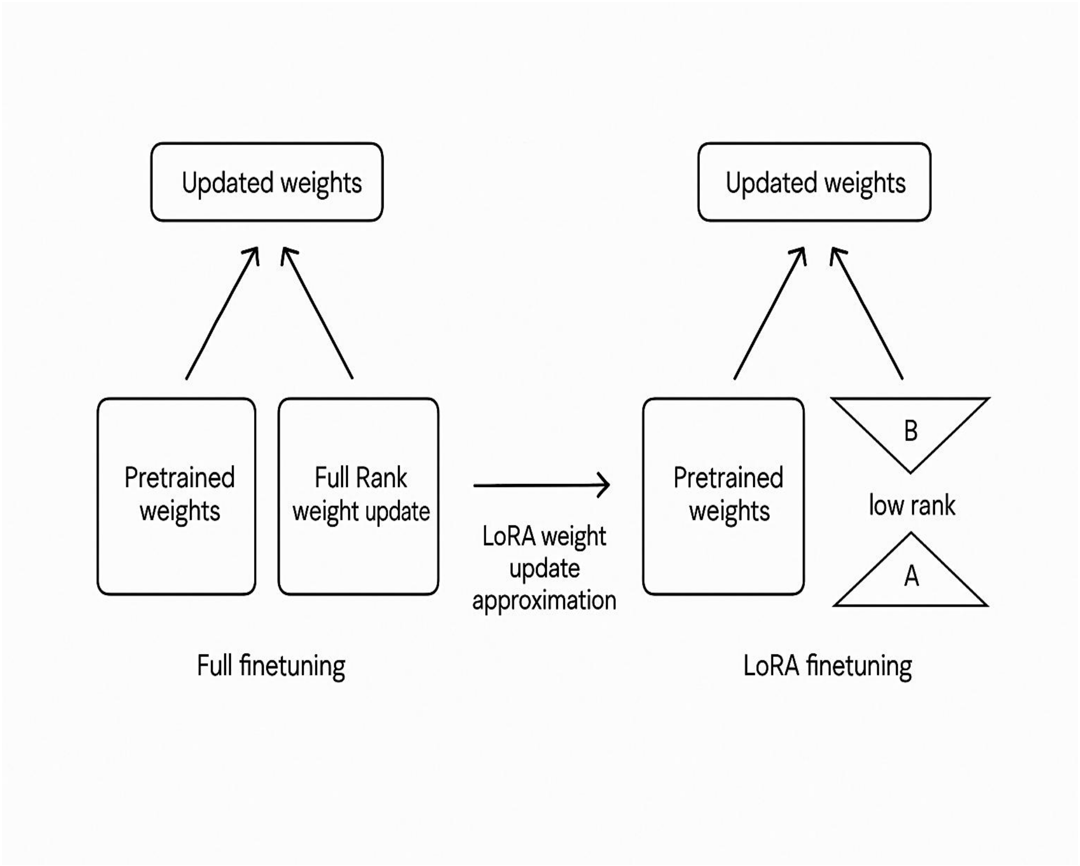


Fig. I. LoRA finetuning [31]

C.2 Quantization for Memory Efficiency

Quantization is a technique used to compress models by reducing the numerical precision of their weights. While models are typically trained using 32-bit floating-point (FP32) numbers, quantization allows them to be represented and processed in lower-precision formats like 16-bit (FP16), 8-bit integers (INT8), or even 4-bit integers (INT4). This dramatically reduces the model's memory footprint (VRAM usage) and can accelerate inference speed.

C.3 QLoRA: Combining PEFT and Quantization for Training

A key challenge is that quantization can degrade performance if applied naively, and performing PEFT on a fully loaded FP32 model can still exceed the memory of consumer GPUs. To solve this, we employed QLoRA (Quantized Low-Rank Adaptation), a technique that brilliantly combines these two methods.

The QLoRA process, as implemented through the Unsloth framework and the bitsandbytes library, works as follows:

- The base DeepSeek model is first quantized to an ultra-low 4-bit precision and loaded into GPU memory. This is the source of the massive memory savings.
- The LoRA adapter matrices are then attached to this quantized base model.
- During training, the 4-bit weights are de-quantized to a higher precision (in our case, 16-bit float) only when needed for forward and backward passes, after which they are discarded from memory. Only the small LoRA adapters are updated in full precision.

This "on-the-fly" de-quantization allows us to achieve the memory savings of a 4-bit model while maintaining the training stability and performance of a 16-bit model.

C.4 Relevance to Our Work

The combination of QLoRA and the Unsloth framework was essential to the success of this research. It enabled us to fine-tune a powerful 8-billion-parameter model on a single, consumer-grade Tesla T4 GPU with just 15 GB of VRAM.

III. MITRE ATT&CK AS A STRUCTURED KNOWLEDGE BASE

The MITRE ATT&CK framework[38] is a comprehensive, structured knowledge base that systematically catalogs real-world adversarial behaviors, including tactics, techniques, sub-techniques, mitigations, and data sources.

ATT&CK organizes cyberattack behavior across multiple domains—Enterprise, Mobile, and ICS—using a three-tiered hierarchy of tactics (adversary goals), techniques (methods), and sub-techniques (specific implementations). This hierarchical structure, grounded in empirical threat intelligence, supports granular analysis of attacker behavior throughout the kill chain. This makes ATT&CK not just a reference tool but a practical operational asset that can be integrated into SIEM, XDR, SOAR, and threat-hunting workflows. Its structured, machine-readable format and broad adoption across the cybersecurity ecosystem make it a cornerstone for building robust, intelligence-driven defense strategies in modern SOC operations.

IV. METHODOLOGY

Our methodology encompasses three primary stages: (1) the construction of a domain-specific, instruction-style dataset grounded in the MITRE ATT&CK framework(Fig. II.); (2) a rigorous validation process to ensure data quality; and (3) the experimental setup for fine-tuning our target LLM.

A.1 Dataset Construction

To create a high-quality corpus for fine-tuning, we developed a structured dataset of question-answer pairs derived from the MITRE ATT&CK Playbooks repository on GitHub [39]. This repository was selected for its practical, analyst-centric guidance on detecting, investigating, and containing real-world adversarial behaviors.

A.1 Automated Data Extraction

A Python script was developed to systematically retrieve, parse, and transform the markdown-formatted playbook content into a structured dataset suitable for instruction tuning. The process involved:

- File Retrieval: All .md files were fetched from the GitHub repository using its REST API.
- Markdown Parsing: The markdownify library was used to convert markdown to plain text. Custom logic then extracted key sections relevant to SOC workflows, such as Log Sources, Key Indicators, Analyst Actions (T1/T2), and Containment Strategies.
- Prompt Engineering: The extracted content was programmatically structured into 4,096 instruction-response pairs. Each pair was formatted to simulate a realistic analyst query using a consistent template (e.g., "What are the key indicators of MITRE technique T1056?"). All entries were structured in the {"instruction": ..., "input": "", "output": ...} format, ensuring compatibility with standard Supervised Fine-Tuning (SFT) pipelines.

The resulting dataset, mitre_instructions_curated.jsonl, served as the domain-specific knowledge base for fine-tuning the DeepSeek R1 Distilled LLaMA 8B model.

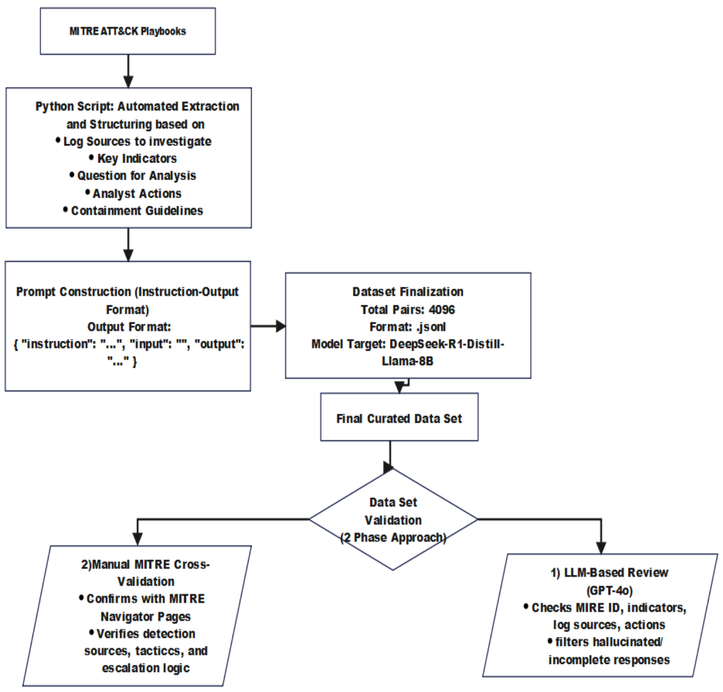

MITRE ATT&CK Playbooks
Python Script: Automated Extraction and Structuring based on
• Log Sources to investigate
• Key Indicators
• Question for Analysis
• Analyst Actions
• Containment Guidelines
Prompt Construction (Instruction-Output Format)
Output Format:
{ "instruction": "...", "input": "", "output": "..." }
Dataset Finalization
Total Pairs: 4096
Format: .jsonl
Model Target: DeepSeek-R1-Distill-Llama-8B
Final Curated Data Set
Data Set Validation (2 Phase Approach)
2)Manual MITRE Cross-Validation
• Confirms with MITRE Navigator Pages
• Verifies detection sources, tacticcs, and escalation logic
1) LLM-Based Review (GPT-4o)
• Checks MIRE ID, indicators, log sources, actions
• filters hallucinated/ incomplete responses

Fig. II. Data Set Creation

B. Dataset Validation
To ensure the factual correctness and operational relevance of the dataset, we implemented a robust, two-phase validation process before fine-tuning.

B.1 LLM-as-a-Judge Pre-Validation
We first used a powerful generalist model (OpenAI's GPT-4o) to perform a "sanity check" on a sample of the dataset. For each Q&A pair, the judge model was prompted to act as an expert reviewer and assess whether the response was factually correct, logically consistent with the question, and aligned with standard SOC practices. This allowed us to quickly flag and correct any programmatic parsing errors, hallucinations from the source material, or ambiguous outputs.

B.2 MITRE ATT&CK Framework Cross-Validation
Next, a manual cross-check was conducted against the official MITRE ATT&CK Navigator. For each technique, we validated key dimensions such as technique ID consistency, recommended data sources, and alignment with procedural examples. A representative case study for T1110 (Brute Force) illustrates this process:
● Technique ID: Correctly identified.
● Log Sources: The dataset correctly listed essential sources like Windows Event Logs (4625), application logs (SSH, RDP), and cloud sign-in logs.
● Key Indicators & Analyst Actions: The generated content aligned well with real-world indicators (e.g., multiple failures from a single IP, off-hours attempts) and standard containment strategies (e.g., account lockouts, MFA enforcement).
While some generated entries lacked exhaustive detail (e.g., specific numerical thresholds for escalation), the validation confirmed that the dataset was semantically sound, factually grounded, and operationally relevant.

C. Experimental Setup and Fine-Tuning
To fine-tune the DeepSeek R1 Distilled LLaMA 8B model on our curated MITRE ATT&CK dataset, we designed a resource-efficient experimental setup leveraging the Unsloth framework and QLoRA. This approach was specifically chosen to demonstrate the feasibility of domain-specific adaptation on accessible, consumer-grade hardware.

C.1 Environment and Model Configuration
The training was conducted on the Kaggle cloud platform, utilizing a dual Tesla T4 GPU environment, each with 14.7 GB of VRAM, running CUDA Toolkit 12.6. We used Unsloth v2025.5.7, a framework that provides significant optimizations for training LLaMA-style models. Unsloth accelerates training and reduces memory usage by patching transformer layers and implementing intelligent gradient offloading, making it ideal for memory-constrained environments.
The base model, unsloth/DeepSeek-R1-Distill-Llama-8B, was loaded with a 4-bit quantization strategy via the bitsandbytes library. For parameter-efficient fine-tuning, we implemented LoRA (Low-Rank Adaptation) with the following key configurations, targeting the model's attention and feed-forward layers:
● LoRA Rank (r): 16
● LoRA Alpha (lora_alpha): 16
● Target Modules: q_proj, k_proj, v_proj, o_proj, gate_proj, up_proj, down_proj
● Trainable Parameters: ~41.9 million (0.52% of total)

C.2 Training Procedure

We utilized the SFTTrainer from the HuggingFace trl library for supervised fine-tuning. The 4,096 instruction-response pairs were formatted using a consistent Alpaca-style prompt template to ensure stable learning. The training was configured with the hyperparameters detailed in Table I.

TABLE I HYPERPARAMETERS USED FOR FINETUNING

Hyperparameter	**Value**
Epochs	3
Batch Size (per device)	2
Gradient Accumulation	8
Learning Rate	1e-4
Optimizer	adamw_8bit
Scheduler	Linear
Max Tokens	2048
Weight Decay	0.01

V. TRAINING DYNAMICS AND PERFORMANCE

The fine-tuning process was monitored using Weights & Biases (W&B) to track training stability, convergence, and resource utilization [40]. The final LoRA adapter weights were saved for subsequent evaluation.

A. Training Convergence and Stability

The model demonstrated stable and efficient convergence over 378 training steps, completed in approximately 2.3 hours. The training loss showed a consistent downward trend, decreasing from an initial value of ~2.95 to a final value of ~1.13, indicating effective learning from the instruction dataset (Fig. III). The gradient norm remained stable within a healthy 0.6–1.2 range, confirming the absence of gradient explosion or vanishing issues (Fig. IV).

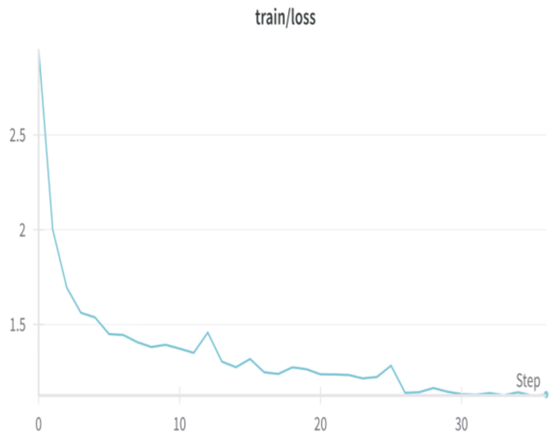


Fig. III. Training Loss Progression

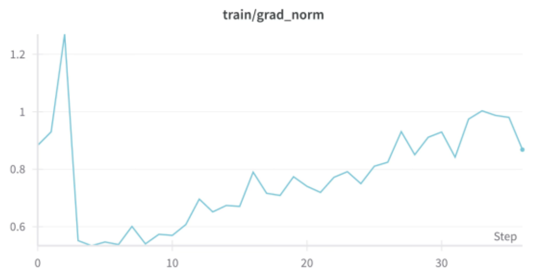


Fig. IV. Gradient Norm Behavior

B. Resource Efficiency

The combination of Unsloth, QLoRA, and an 8-bit optimizer proved highly effective for resource-constrained training. The primary GPU maintained a power utilization rate of 85–95% (Fig. V)while operating at a stable temperature of ~75°C, well within thermal safety margins(Fig. VI).

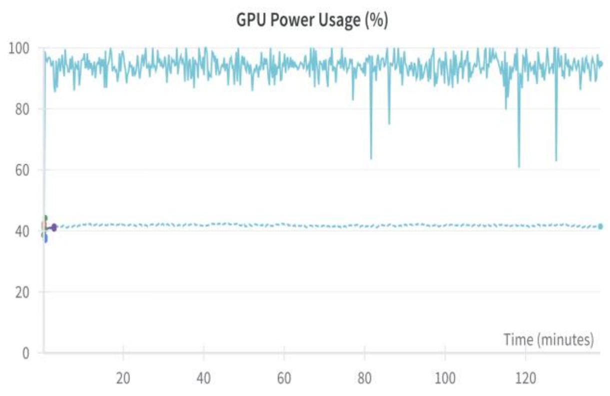


Fig. V. GPU Power Usage (%)

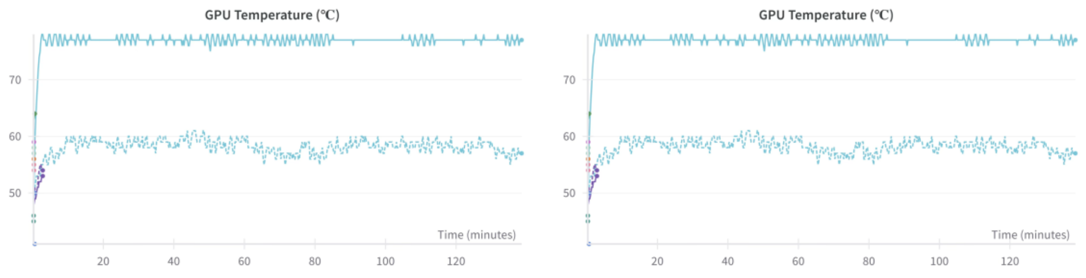


Fig. VI. GPU Temperature Monitoring

VI. EVALUATION FRAMEWORK

To systematically assess the practical utility of our fine-tuned model, we designed a multi-phase evaluation framework comparing it against the untuned DeepSeek R1 base model. The framework was structured to measure not only factual recall but also applied reasoning, performance efficiency, and overall value as a SOC analyst co-pilot.

A. Phase 1: Foundational Knowledge and Performance Benchmarking

The initial phase focused on quantitative metrics to validate the success of the fine-tuning process and establish a performance baseline.

● **Technique Identification (Exact Match):** We tested both models on a set of 25 direct Q&A prompts from our training data (e.g., "What is the MITRE ID for Data from Local System?"). Performance was measured by Exact Match Accuracy, which calculates the percentage of responses that exactly match the ground-truth MITRE ID.

● **Performance Benchmarking:** We measured two key efficiency metrics for both models during inference:

○ **Inference Latency:** The average time in seconds required to generate a complete response to a standard analyst query.

○ **VRAM Footprint:** The amount of GPU memory in gigabytes consumed by the loaded 4-bit quantized model.

B. Phase 2: Applied Reasoning Test (The "Analyst Co-pilot" Scenario)

The core of our evaluation was a realistic test of applied reasoning. We prompted both the untuned "generalist" and the fine-tuned "specialist" to analyze 11 real-world security alerts and generate a structured co-pilot response.

Crucially, our initial experiments with simple, direct prompts (e.g., "Create a plan for this alert") revealed a critical failure mode in the fine-tuned model. While factually knowledgeable, it was "brittle" and unable to apply its knowledge to a complex, multi-step task, often resulting in output collapse (nan) or non-contextual responses.

To overcome this, we developed a more sophisticated inference strategy: an "Example-Guided Synthesis" (One-Shot Learning) prompt. This single, comprehensive prompt provided each model with:

1. A high-quality, expert-written example of an ideal analysis.
2. The specific details of the new security alert.
3. A clear, structured task: generate a ## Threat Explanation and a 5-step ## Investigation Plan by following the format of the example.

This methodology allowed us to test the models' ability to perform complex synthesis when properly guided, mirroring a more advanced and realistic human-AI interaction.

C. Phase 3: LLM-as-a-Judge (Qualitative Evaluation)

To score the open-ended outputs from the co-pilot test, we used a comparative LLM-as-a-Judge (LLMJ) framework. A powerful, impartial third-party model (OpenAI's GPT-4o) was prompted to act as an expert cybersecurity analyst. For each of the 11 alerts, the judge was presented with the alert context and the responses from both the base and fine-tuned models.The judge was instructed to score each response based on three core criteria essential for a SOC analyst:

● Factual Accuracy: Does the explanation correctly identify the MITRE technique and avoid hallucinations?

● Contextual Relevance: Is the response tailored to the specific details in the alert (e.g., IPs, hostnames, logs)?

● Actionability: Is the investigation plan composed of concrete, specific steps an analyst could actually perform?

Finally, the judge was required to provide a qualitative analysis of each model's strengths and weaknesses and declare a "Final Winner" for that specific alert. The complete prompt template used for this evaluation is detailed below.

" " "

```
You are an expert cybersecurity analyst and a meticulous grader for a research paper. Your task is to perform a head-to-head comparison of two AI-generated responses to a security alert. Your judgment should be based on which response would be more valuable, accurate, and actionable for a real Tier 1 SOC analyst.
Please assess each AI response individually first, then declare an overall winner.
**Evaluation Criteria:**
-   **Factual Accuracy:** Is the threat explanation correct? Does it avoid hallucinations?
-   **Contextual Relevance:** Is the response tailored to the specific details in the alert (IPs, hostnames, logs)?
-   **Actionability:** Is the investigation plan composed of concrete, specific steps an analyst could actually perform?
---
### **EVALUATION ITEM**
**[SECURITY ALERT DETAILS]**
{{alert_context f}}
---
**[RESPONSE FROM BASE MODEL]**
{{base_model_output}}
---
**[RESPONSE FROM FINE-TUNED MODEL]**
{{finetuned_model_output}}
---
### **YOUR JUDGEMENT**
Please provide your evaluation in the following structured format:
**1. Analysis of Base Model's Response:**
   - **Strengths:** (e.g., "Well-written and structured," "Provides a logical plan.")
   - **Weaknesses:** (e.g., "Factually incorrect explanation," "Hallucinated the MITRE technique," "Plan is too generic.")
   - **Overall Quality Score (1-5):**
**2. Analysis of Fine-Tuned Model's Response:**
   - **Strengths:** (e.g., "Factually accurate explanation," "Plan is relevant to the alert.")
   - **Weaknesses:** (e.g., "Explanation is not conversational," "Plan could be more specific.")
   - **Overall Quality Score (1-5):**
**3. Head-to-Head Comparison & Final Verdict:**
   - **Which model's response is more useful and trustworthy for a SOC analyst?**
   - **Brief Justification:** (Explain *why* one is better. Focus on the impact of factual accuracy vs. fluency.)
   - **Final Winner (Choose one):**
     [ ] Base Model is Significantly Better
     [ ] Base Model is Slightly Better
     [ ] Both are Comparable / Have different strengths
     [ ] Fine-Tuned Model is Slightly Better
     [ ] Fine-Tuned Model is Significantly Better
” ” ”
```

Prompt. I. LLM-as-a-Judge Comparative Evaluation Prompt

VII. RESULTS AND ANALYSIS

Our multi-phase evaluation revealed a nuanced and critical set of trade-offs between the untuned base model and our fine-tuned specialist. While fine-tuning was essential for factual accuracy, the base model occasionally demonstrated superior, albeit flawed, reasoning capabilities.

A. Foundational Performance: The Undisputed Value of Fine-Tuning

Our initial benchmarks confirmed the success and efficiency of the fine-tuning process. On a direct Q&A task for technique identification, the fine-tuned model achieved 84% exact match accuracy, while the untuned base model scored 0%, consistently hallucinating incorrect MITRE IDs as shown in Fig VII and Fig VIII. Furthermore, the fine-tuned model demonstrated a ~3x speed improvement in inference latency (11.09s vs. 33.11s) for a negligible increase in VRAM footprint as detailed in Table II. These results establish that fine-tuning is highly effective for embedding domain knowledge and optimizing performance.

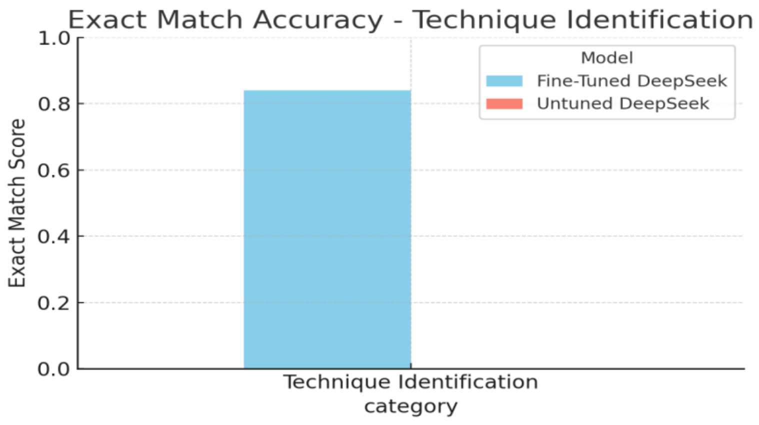


Fig. VII. Exact Match Accuracy Evaluation

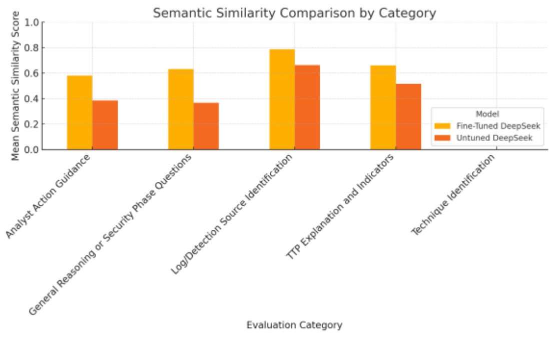


Fig. VIII. Semantic Similarity Evaluation

TABLE II QUANTITATIVE PERFORMANCE AND EFFICIENCY BENCHMARK

Metric	**Untuned Base Model**	**Fine-Tuned Model**	**Advantage of Fine-Tuning**
Average Inference Latency	33.11 seconds	**11.09 seconds**	**~3x Faster**
Model VRAM Footprint	5.99 GB	**6.16 GB**	Negligible (+2.8%)
Factual Accuracy	Very Low (Prone to Hallucination)	**High** (Grounded in Training Data)	**Qualitatively Superior**

B. Co-pilot Performance: The Indispensable Role of Prompt Engineering

B.1 Initial Test: Programmatic Chain of Thought

Our first evaluation used a "Programmatic Chain of Thought" (a “zero-shot” approach) approach. This two-step process first prompted the models to recall their knowledge about a MITRE technique and then asked them to synthesize an investigation plan based on that recalled knowledge. This initial test immediately revealed a critical divergence between the two models.

● The Untuned Base Model consistently hallucinated the factual basis of the MITRE technique. While its investigation plans were often logically structured, they were based on a fundamentally incorrect premise, making them dangerously misleading.

● The Fine-Tuned Model, while factually accurate in its knowledge recall, proved to be a "brittle specialist." It often failed to synthesize its knowledge into a human-friendly format, producing either non-contextual data dumps or, in cases of slight prompt mismatch, complete output collapse (nan).

This confirmed that specialized knowledge alone is insufficient for practical application.

To overcome this, we developed a more sophisticated inference strategy: **"Example-Guided Synthesis" (One-Shot Learning).** Instead of a simple query, we provided each model with a single, comprehensive prompt containing a high-quality example of the desired output. This single example acted as a powerful guide, teaching the model how to structure its response and apply its knowledge to the specific alert context. This method yielded a dramatic improvement in the usability and quality of both models' outputs, forming the basis for our head-to-head evaluation.

B.2 Improved Method: Example-Guided Synthesis (One-Shot Learning)

To address the brittleness of the fine-tuned model, we implemented a more sophisticated "Example-Guided Synthesis" (One-Shot) prompt. This single, comprehensive prompt provided each model with a high-quality example of the desired output, guiding it to perform both threat explanation and plan generation in a structured manner. This method yielded a dramatic improvement in the usability of both models' outputs.

1. **Qualitative Analysis: The Brittle Specialist vs The Fluent Hallucinator:**

Our case-by-case analysis highlighted a critical trade-off between the two models.

- **The Untuned Base Model (The Fluent Hallucinator):** "The base model consistently demonstrated superior contextual reasoning. For example, in the T1014 alert, it correctly created a plan focused on investigating the hidden process 2508. However, this reasoning was built upon a foundation of catastrophic factual hallucination, as it incorrectly identified T1014 as 'OS Credential Dumping.' This makes the base model dangerously unreliable."
- **The Fine-Tuned Model (The Brittle Specialist):** Its primary strength is its factual accuracy. It never hallucinated a MITRE technique and its threat explanations were always grounded in its specialized training. Its investigation plans, while sometimes less exhaustive than the base model's, were consistently relevant, actionable, and free of the technical errors (e.g., incorrect commands) that plagued the base model's output.

2. **Quantitative Findings:** Fine-Tuning Delivers Superior Quality

The qualitative superiority demonstrated in the case studies is further substantiated by a quantitative analysis of the LLM-as-a-Judge (LLMJ) verdicts across all 11 evaluation scenarios. When judged on a 5-point scale for overall quality, the fine-tuned model achieved an average score of 4.09, significantly outperforming the base model's score of 3.27.

The head-to-head comparison further reinforces this conclusion (Table III). In a direct final verdict, the judge deemed the fine-tuned model's response superior in 72.7% of cases, compared to just 18.2% for the base model.

TABLE III FINAL VERDICT OF LLM ON FINE-TUNED AND UNTUNED MODEL

Final Verdict	**Count**	**Percentage**
Fine-Tuned Model Wins (Slightly or Significantly)	8	72.7%
Base Model Wins (Slightly)	2	18.2%
Comparable	1	9.1%
Total	11	100%

1.Case Study: Comparative Output using Example-Guided Synthesis

The following tables present a direct, side-by-side comparison of the responses generated by the untuned and fine-tuned models when guided by the one-shot prompt.

Ingested Alert:

{"timestamp": "2024-05-21T10:05:15.123+0000", "rule": {"level": 11, "description": "Possible kernel level rootkit", "id": "521", "mitre": {"id": ["T1014"], "tactic": ["Defense Evasion"], "technique": ["Rootkit"]}, "firedtimes": 1, "mail": false, "groups": ["ossec", "rootcheck"]}, "agent": {"id": "ALU-003", "name": "ALU-ORDPROC-PROD-01", "ip": "10.60.5.25", "labels": {"group": "aluskort-prod"}}, "manager": {"name": "wazuh-manager-alus"}, "id": "1716285915.9646311", "full_log": "Process '2508' hidden from /proc. Possible kernel level rootkit.", "decoder": {"name": "rootcheck"}, "data": {"title": "Process '2508' hidden from /proc."}, "location": "rootcheck"}

TABLE IV COMPARATIVE OUTPUT FOR MITRE ID-T1014 (ROOTKIT) ALERT

Model Type	**Generated Co-pilot Response**
Untuned Base Model	## **Threat Explanation** The log snippet suggests that a process named '2508' has been hidden within /proc, which typically indicates the presence of a kernel-level rootkit..**Investigation Plan** 1. **Kernel Inspection:** Inspect the /proc directory for additional hidden processes using tools like ls -lO. 2. **Memory Dump Analysis:** Capture a memory dump using tools such as dumpmem or mman.
Fine-Tuned Model	## **Threat Explanation** The presence of process ID 2508 being hidden suggests potential rootkit activity. Rootkits typically hide processes by manipulating system calls such as hidepid() which can make them undetectable... **Investigation Plan** 1. **Isolate Affected System:** Quarantine the host ALU-ORDPROC-PROD-01 to prevent lateral movement..2. **Memory Analysis:** Perform forensic imaging of memory to capture detailed process information...4. **Rootkit Scanning Tools:** Utilize advanced malware detection tools designed to detect rootkits...
Expert Judge's Verdict	**Fine-Tuned Model is Significantly Better.** "The fine-tuned model produces a more accurate, context-aware, and operationally realistic response. It avoids tool hallucinations and presents a clean investigation path. While the base model attempts

	broader coverage, its technical inaccuracies could confuse a Tier 1 analyst."

Ingested Alert:

{"timestamp": "2024-05-21T10:10:30.456+0000", "rule": {"level": 10, "description": "High amount of POST requests in a small period of time (likely bot).", "id": "31533", "mitre": {"id": ["T1498"], "tactic": ["Impact"], "technique": ["Network Denial of Service"]}, "frequency": 8, "firedtimes": 1, "mail": false, "groups": ["web", "appsec", "attack"], "pci_dss": ["6.5", "11.4"], "gdpr": ["IV_35.7.d"], "nist_800_53": ["SA.11", "SI.4"], "tsc": ["CC6.6", "CC7.1", "CC8.1", "CC6.1", "CC6.8", "CC7.2", "CC7.3"]}, "agent": {"id": "ALU-001", "name": "ALU-WEB-PROD-01", "ip": "10.50.1.10", "labels": {"group": "aluskort-prod"}}, "manager": {"name": "wazuh-manager-alus"}, "id": "1716286230.10158622", "previous_output": "198.51.100.25 - 10.50.1.10 80 POST /cart/add - 200 - AluskortBot/1.0\n198.51.100.25 - 10.50.1.10 80 POST /cart/add - 200 - AluskortBot/1.0\n198.51.100.25 - 10.50.1.10 80 POST /cart/add - 200 - AluskortBot/1.0\n198.51.100.25 - 10.50.1.10 80 POST /cart/add - 200 - AluskortBot/1.0\n198.51.100.25 - 10.50.1.10 80 POST /cart/add - 200 - AluskortBot/1.0\n198.51.100.25 - 10.50.1.10 80 POST /cart/add - 200 - AluskortBot/1.0\n198.51.100.25 - 10.50.1.10 80 POST /cart/add - 200 - AluskortBot/1.0", "full_log": "198.51.100.25 - 10.50.1.10 80 POST /cart/add - 200 - AluskortBot/1.0", "decoder": {"name": "web-accesslog"}, "data": {"protocol": "POST", "srcip": "198.51.100.25", "id": "200", "url": "/cart/add"}, "location": "C:\\inetpub\\logs\\LogFiles\\W3SVC1\\u_ex240521.log"}

TABLE V COMPARATIVE OUTPUT FOR MITRE ID-T1498

Model Type	Generated Co-pilot Response
Untuned Base Model	## **Threat Explanation** The high volume of POST requests originating from 198.51.100.25... suggests automated behavior typical of bots. This could indicate a DDoS attack, scraping activity, or unauthorized automation... **Investigation Plan** 1. **Identify the Bot Type:** Use tools like browser developer tools to inspect the user-agent string (AluskortBot/1.0)...2. **Check Request Payloads:** Analyze the POST data being sent to /cart/add...
Fine-Tuned Model	## **Threat Explanation** The detected activity suggests automated scanning behavior typical of bots. The high volume of POST requests could indicate attempts to manipulate data within the application... **Investigation Plan** 1. **Identify Bots:** Use tools like Bot Defender to detect patterns consistent with botnet traffic originating from 198.51.100.25. 2. **Review Access Logs:** Analyze recent changes in database records related to customer accounts accessed by this IP...3.. **Block Malicious IPs:** Temporarily isolate the identified IP address 198.51.100.25 using firewalls...
Expert Judge's Verdict	**Base Model is Slightly Better.** "The base model delivers a broader and more technically grounded plan. It links directly to OAuth misconfigurations... and provides a robust mitigation checklist. While the fine-tuned model offers a tighter narrative... it sacrifices some important technical investigative angles."

B.3 Analysis of Example-Guided Results

The one-shot prompting strategy successfully guided both models to produce structured, context-aware, and actionable outputs. This confirms that for complex, multi-step tasks, providing a clear example is a highly effective inference strategy. While the fine-tuned model was superior in the majority of cases, it is noteworthy that in the T1498 (Denial of Service) alert, the judge favored the base model's response for its greater technical depth and broader investigative scope. This highlights that the fine-tuned model's knowledge, while accurate, may be constrained by the scope of its training data.

VIII. DISCUSSION

A. The Specialist's Dilemma: Navigating the Trade-Off between Reliability, Context and Creativity

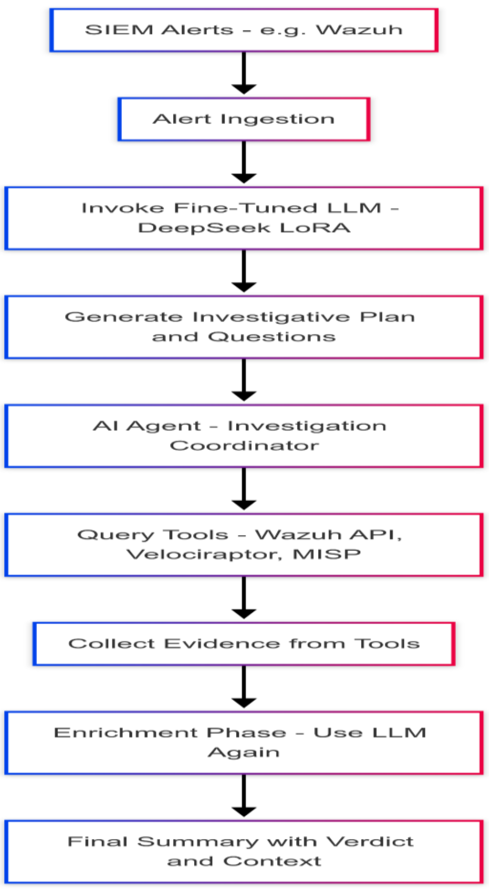
Our research uncovered a critical dynamic we term the "Specialist's Dilemma," a multi-faceted trade-off between the fine-tuned model and its untuned base. This dilemma extends beyond mere accuracy into the very nature of their reasoning. We identified two distinct personas:

1. **The "Imaginative Generalist" (Untuned Base Model):** This model excelled at contextual reasoning and creative depth. Even in zero-shot scenarios, it effortlessly wove specific alert details (IPs, processes) into its investigation plans. Its proposed steps were often more intricate and demonstrated a broader, more "imaginative" approach to problem-solving. However, this creativity was built on a foundation of catastrophic factual inaccuracy. It hallucinated non-existent MITRE techniques and generated plausible-sounding fiction, making it dangerously unreliable for any mission-critical security task.
2. **The "Disciplined Specialist" (Fine-Tuned Model):** Fine-tuning created a model with unwavering factual trustworthiness. It completely eliminated hallucinations, grounding every response in the established MITRE ATT&CK

ontology. This reliability, however, came at the cost of zero-shot creativity and context. The model defaulted to generic, "by-the-book" responses, failing to apply its deep knowledge to the specific alert details. It became a perfect knowledge base but an imperfect reasoning engine.
This reveals a more profound trade-off: the unconstrained, creative reasoning of a generalist versus the reliable, structured knowledge of a specialist.

A.1 Resolving the Dilemma: The Power of Guided Application
The solution lies in how the specialist model is prompted. By providing a one-shot example, we effectively gave the "Disciplined Specialist" a blueprint for applied reasoning. This guidance unlocked its ability to integrate specific alert context while retaining its core factual accuracy.
While this technique restored the model's ability to be context-aware, it's important to acknowledge that it may not fully replicate the unconstrained "creativity" of the base model. The fine-tuned model, even when guided, is still biased toward its structured training data.
The optimal architecture, therefore, is a system that strategically combines the strengths of both worlds. This hybrid approach consciously trades the base model's boundless but dangerous creativity for the fine-tuned model's mission-critical reliability.

B. Practical Implications: The Case for Open-Source integration in the SOC
We propose an architecture (Fix. X.) where the fine-tuned LLM is integrated into a security pipeline (e.g., with Wazuh, TheHive, or MISP) to automatically enrich alerts with structured labels, cluster related events, and generate concise, analyst-friendly summaries.
This would transform a standard alert into an enriched, actionable intelligence product. For example, for the T1498 alert in our evaluation, the system could generate the following summary for the analyst:
“This alert maps to MITRE Technique T1498: Network Denial of Service. The behavior involves a high frequency of POST requests to /cart/add from source IP 198.51.100.25 targeting web asset ALU-WEB-PROD-01. The repetitive, automated pattern, identified by the user-agent AluskortBot/1.0, suggests bot-driven application-layer DoS activity aimed at overwhelming the shopping cart endpoint.”
While the full implementation of this pipeline was outside the scope of our current study, it represents a viable and high-impact deployment pathway. Future work should focus on deploying this architecture in an operational testbed to measure improvements in key SOC metrics such as Mean Time to Detect (MTTD) and Mean Time to Respond (MTTR), as well as reductions in false positive rates. By grounding a privacy-preserving LLM in a structured ontology like MITRE ATT&CK, organizations can build powerful, cost-effective, and explainable AI assistance for their cyber defense workflows.


SIEM Alerts - e.g. Wazuh
Alert Ingestion
Invoke Fine-Tuned LLM - DeepSeek LoRA
Generate Investigative Plan and Questions
AI Agent - Investigation Coordinator
Query Tools - Wazuh API, Velociraptor, MISP
Collect Evidence from Tools
Enrichment Phase - Use LLM Again
Final Summary with Verdict and Context
Fig. IX. Proposed AI SOC architecture

C. Economic and Privacy Implications: A Cost Benefit Analysis

Beyond performance, the choice between deploying a locally-hosted, open-source model versus using a commercial API from a large provider has profound implications for cost, data privacy, and operational control. To quantify this trade-off, we conducted a cost-benefit analysis comparing our fine-tuned DeepSeek model against a top-tier commercial alternative (e.g., GPT-4.1).

Our approach, which leverages a fine-tuned open-source model, directly addresses these challenges. By enabling local deployment on consumer-grade hardware, it guarantees that all sensitive data remains within the organizational boundary. Commercial LLM APIs typically operate on a per-token pricing model. Based on the "Example-Guided Synthesis" prompt used in our evaluation, we estimate an average of approximately 500 input tokens and 500 output tokens per alert analysis. Using the pricing for a model like GPT-4.1 (approximately \$3.00 per 1M input tokens and \$12.00 per 1M output tokens [41]), the cost per alert is calculated as follows:

Cost per Alert = (500/1M × \$3.00) + (500/1M × \$12.00) = \$0.0075

While seemingly small, this cost scales linearly with alert volume. For a mid-sized SOC handling 100,000 alerts per month, this translates to a recurring operational expense of approximately \$750 per month or \$9,000 annually. For a large MSSP, these costs could easily run into the tens of thousands per month."It is important to note that these prices are subject to change but serve as a representative example of the usage-based cost model inherent to commercial APIs.

IX. CONCLUSION AND FUTURE WORK

In this study, we investigated the efficacy of fine-tuning an open-source LLM for specialized cybersecurity analysis. Our findings demonstrate that while domain-specific fine-tuning is an essential step for building a trustworthy AI co-pilot—successfully eliminating the dangerous factual hallucinations present in the base model—it is not a complete solution. The fine-tuning process produces what we term a "brittle specialist": a model that is factually accurate but whose knowledge is static and difficult to apply to novel contexts. We demonstrated that this brittleness can be partially mitigated through sophisticated, "Example-Guided" (one-shot) prompting. However, this still relies on the model's static, internalized knowledge. In the ever-changing landscape of cybersecurity, where new threats, vulnerabilities, and techniques emerge daily, a model with a fixed knowledge cut-off is fundamentally inadequate for long-term production use. Based on these findings, we conclude that the optimal architecture for a production SOC must decouple knowledge retrieval from reasoning. In future we will focus on implementing a Retrieval-Augmented Generation (RAG) system. A RAG architecture would leverage a reliable, external vector database—populated with our MITRE ATT&CK corpus and continuously updated with new threat intelligence—to retrieve perfect, timely context. This context would then be provided to a powerful generalist model whose sole task is synthesis. This approach combines the factual reliability of a database with the superior contextual reasoning of a general-purpose LLM, offering a more scalable, maintainable, and ultimately more effective solution for deploying trustworthy AI in the Security Operations Center.

REFERENCES

[1] Ashish, V., 2017. Attention is all you need. Advances in Neural Information Processing Systems, 30, p.I.

[2] Singh, B., 2024. Introduction to Large Language Models. In Building Applications with Large Language Models: Techniques, Implementation, and Applications (pp. 1-25). Berkeley, CA: Apress.

[3] Yenduri, G., Srivastava, G., Maddikunta, P.K.R., Jhaveri, R.H., Wang, W., Vasilakos, A.V. and Gadekallu, T.R., 2023. Generative pre-trained transformer: A comprehensive review on enabling technologies, potential applications, emerging challenges, and future directions. arXiv preprint arXiv:2305.10435.

[4] Yao, Y., Duan, J., Xu, K., Cai, Y., Sun, Z. and Zhang, Y., 2024. A survey on large language model (LLM) security and privacy: The good, the bad, and the ugly. High-Confidence Computing, p.100211.

[5] Yigit, Y., Buchanan, W.J., Tehrani, M.G. and Maglaras, L., 2024. Review of generative AI methods in cybersecurity. arXiv preprint arXiv:2403.08701.

[6] Pan, J., Liang, W.S. and Yidi, Y., 2024, May. Raglog: Log anomaly detection using retrieval augmented generation. In 2024 IEEE World Forum on Public Safety Technology (WFPST) (pp. 169-174). IEEE.

[7] Alahmadi, B.A., Axon, L. and Martinovic, I., 2022, August. 99% False Positives: A Qualitative Study of SOC Analysts' Perspectives on Security Alarms. In Proceedings of the 31st USENIX Security Symposium (USENIX Security), Boston, MA, USA (pp. 10-12).

[8] Yang, G., Tang, C. and Liu, X., 2022. DualAC2NN: Revisiting and Alleviating Alert Fatigue from the Detection Perspective. Symmetry, 14(10), p.2138.

[9] Hassan, W.U., Guo, S., Li, D., Chen, Z., Jee, K., Li, Z. and Bates, A., 2019, February. Nodoze: Combatting threat alert fatigue with automated provenance triage. In Network and Distributed Systems Security Symposium.

[10] Bryant, B.D. and Saiedian, H., 2020. Improving SIEM alert metadata aggregation with a novel kill-chain based classification model. Computers & Security, 94, p.101817.

[11] Zhong, C., Yen, J., Liu, P. and Erbacher, R.F., 2019. Learning from experts' experience: Toward automated cyber security data triage. IEEE Systems Journal, 13(1), 603-614. doi:10.1109/JSYST.2018.2828832.

[12] González-Granadillo, G., González-Zarzosa, S. and Diaz, R., 2021. Security information and event management (SIEM): Analysis, trends, and usage in critical infrastructures. Sensors, 21(14), p.4759.

[13] Cinque, M., Della Corte, R. and Pecchia, A., 2020. Contextual filtering and prioritization of computer application logs for security situational awareness. Future Generation Computer Systems, 111, 668-680.

[14] Bartos, V., Zadnik, M., Habib, S.M. and Vasilomanolakis, E., 2019. Network entity characterization and attack

prediction. Future Generation Computer Systems, 97, 674-686.
[15] Mudgal, P. and Wouhaybi, R., 2023, August. An assessment of ChatGPT on log data. In International Conference on AI-Generated Content (pp. 148-169). Singapore: Springer Nature Singapore.
[16] Deka, P., Rajapaksha, S., Rani, R., Almutairi, A. and Karafili, E., 2024, November. Attacker: Towards enhancing cyber-attack attribution with a named entity recognition dataset. In International Conference on Web Information Systems Engineering (pp. 255-270). Singapore: Springer Nature Singapore.
[17] Cheng, Y., Bajaber, O., Tsegai, S.A., Song, D. and Gao, P., 2024. CTINEXUS: Leveraging optimized LLM in-context learning for constructing cybersecurity knowledge graphs under data scarcity. arXiv preprint arXiv:2410.21060.
[18] Wahréus, J., Hussain, A.M. and Papadimitratos, P., 2025. CySecBench: Generative AI-based cybersecurity-focused prompt dataset for benchmarking large language models. arXiv preprint arXiv:2501.01335.
[19] Ghimire, A., Ghajari, G., Gurung, K., Sah, L.K. and Amsaad, F., 2025. Enhancing cybersecurity in critical infrastructure with LLM-assisted explainable IoT systems. arXiv preprint arXiv:2503.03180.
[20] Si, S., Wu, Y., Tang, L., Zhang, Y., Wosik, J. and Su, Q., 2024. Evaluating the performance of ChatGPT for spam email detection. arXiv preprint arXiv:2402.15537.
[21] Shafee, S., Bessani, A. and Ferreira, P.M., 2024. Evaluation of LLM chatbots for OSINT-based cyber threat awareness. arXiv preprint arXiv:2401.15127.
[22] Zuo, F., Rhee, J. and Choe, Y.R., 2025. Knowledge transfer from LLMs to provenance analysis: A semantic-augmented method for APT detection. arXiv preprint arXiv:2503.18316.
[23] Deng, J., Li, X., Chen, Y., Bai, Y., Weng, H., Liu, Y., Wei, T. and Xu, W., 2024. RACONTEUR: A knowledgeable, insightful, and portable LLM-powered shell command explainer. arXiv preprint arXiv:2409.02074.
[24] Gandhi, P.A., Wudali, P.N., Amaru, Y., Elovici, Y. and Shabtai, A., 2025. SHIELD: APT detection and intelligent explanation using LLM. arXiv preprint arXiv:2502.02342.
[25] Ahmed, S., Rahman, A.M., Alam, M.M. and Sajid, M.S.I., 2025, January. SPADE: Enhancing adaptive cyber deception strategies with generative AI and structured prompt engineering. In 2025 IEEE 15th Annual Computing and Communication Workshop and Conference (CCWC) (pp. 01007-01013). IEEE.
[26] Hays, S. and White, J., 2024. Using LLMs for tabletop exercises within the security domain. arXiv preprint arXiv:2403.01626.
[27] Prapty, R.T., Kundu, A. and Iyengar, A., 2024. Using retriever-augmented large language models for attack graph generation. arXiv preprint arXiv:2408.05855.
[28] GitHub link: https://github.com/deepseek-ai/DeepSeek-R1/blob/main/DeepSeek_R1.pdf
[29] Hugging Face library link: https://huggingface.co/deepseek-ai/DeepSeek-R1#usage-recommendations
[30] Han, Z., Gao, C., Liu, J., Zhang, J. and Zhang, S.Q., 2024. Parameter-efficient fine-tuning for large models: A comprehensive survey. arXiv preprint arXiv:2403.14608.
[31] Wang, L., Chen, S., Jiang, L., Pan, S., Cai, R., Yang, S. and Yang, F., 2024. Parameter-efficient fine-tuning in large models: A survey of methodologies. arXiv preprint arXiv:2410.19878.
[32] Xu, L., Xie, H., Qin, S.Z.J., Tao, X. and Wang, F.L., 2023. Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment. arXiv preprint arXiv:2312.12148.
[33] Hugging Face link: https://huggingface.co/docs/transformers/main_classes/quantization
[34] Hugging Face link: https://huggingface.co/docs/hub/en/gguf
[35] Hugging Face link: https://huggingface.co/docs/transformers/main/en/quantization/gptq
[36] GitHub link: https://github.com/bitsandbytes-foundation/bitsandbytes
[37] GitHub link: https://github.com/unslothai/unsloth
[38] Available online: https://attack.mitre.org/matrices/enterprise/
[39] GitHub link: https://github.com/CodeByHarri/MITRE-ATT_CK-Playbooks
[40] GitHub link: https://github.com/wandb/wandb
[41] Available online – API Pricing: https://openai.com/api/pricing/